

**EXHIBIT F**

## A Method to Detect Intrusive Activity in a Networked Environment<sup>1</sup>

*L.T. Heberlein, K.N. Levitt, B. Mukherjee*

Computer Security Laboratory  
Division of Computer Science  
University of California  
Davis, Ca. 95616

### ABSTRACT

Intrusive activity is occurring on our computer systems, and the need for intrusion detection has been demonstrated. This paper discusses some of the benefits and drawbacks of trying to detect the intrusive activity by analyzing network traffic. A general solution, based on detecting and analyzing abstract objects, is formulated. Finally, results from applying the solution are presented.

### 1. Introduction

Computers are the targets of attacks [3]. Reports appear in the media almost weekly about outsiders breaking into computers, employees misusing computers, and rogue viruses and worms penetrating computer systems. Incidents such as the internet worm of 1988 [3], the Wank worm [3], and the Netherland hackers have gained international recognition, and they serve to emphasize the vulnerability of computer systems around the world.

These reported incidents are cases of intrusive activity in our computer systems. Intrusive activity can be defined as any attempt which, if successful, will result in one of the following:

- disclosure of information against the wishes of the owner of the information
- modification of information against the wishes of the owner of the information
- denial of the use of services by legitimate users of the system
- use of resources against the wishes of the system's owner (e.g. disk or CPU)

The first three bulleted items are discussed in [4]. The last bulleted item, the stealing of resources, covers actual observed activity which did not fit easily into the three previous categories. For example, using our network security monitor (NSM) [8], we have observed an intruder use a system to crack password files. The intruder was not interested in either looking at existing information on the system, modifying information on the system, or denying resources to legitimate user. The intruder simply used the CPU, when it was idle, to crack passwords.

Authentication and access control mechanisms are designed to guard against intrusive activity; however, these mechanisms have not been wholly successful. Failure of these mechanisms is due in part to the ease by which passwords can be compromised, failure by system administrators and users to properly use the access control mechanisms, poor operating system designs, and flawed operating system implementations (i.e., bugs).

The failures of authentication and access control mechanisms are compounded by the decentralization of computer systems and the increased access to a computer system by computer networks. The decentralization of computer systems is the movement away from a single mainframe computer to multiple workstations and personal computers. The movement is fueled by the increasing power and decreasing costs of workstations and personal computers. The result of decentralization is a type of computer system which is administered by people, usually the user community, with little or no formal training in system administration or computer security. This in turn results in a greater chance for poorly configured authentication and access control mechanisms.

Connecting a computer to a network also increases the chances of intrusive activity occurring on that computer since this process increases the number of people who can potentially access it. Connecting a computer to a network provides a path to that computer for every user with access to the network. If the

<sup>1</sup> This work is supported in part by Lawrence Livermore National Laboratory

network is part of the internet, essentially everyone with access to a telephone has a path to that computer.

With the realization that current authentication and access control mechanisms have not provided adequate security against intrusive behavior, institutions which use computers and computer networks have become interested in detecting the intrusive activity which is occurring. If an intrusion can be detected, an institution can at least know from where intrusive activity is coming, how the activity is being perpetrated (and therefore, hopefully how to stop it), and what data have been compromised.

In the summer of 1988, University of California at Davis and Lawrence Livermore National Laboratory began an effort to detect intrusive activity on a network of heterogeneous computer systems. A brief overview of this effort is presented in section two. Sections three and four present the mechanisms by which our monitor detects intrusive activity. And section five presents some of the results of our efforts as well as directions for future research.

## 2. Network Monitor

Intrusion detection systems examine available sources of information about the various operations in a computing system to determine if intrusive activity is occurring. The main source of information for most intrusion detection system is the audit trails generated by the operating system. Although the audit-trail-based analysis has provided a measure of success, a number of limitations exist with this method. First, audit trails traditionally do not provide much of the information necessary to perform security analysis. This is due in part to the historical purpose of audit trail collection - the billing of customers. Second, audit trails tend to be system specific. Each operating system provides a different set of information in a different format. An intrusion detection system designed to work on the Multics operating system's audit trails would need a great deal of restructuring to operate on another operating system's audit trails. Third, the collection of audit trails is expensive in terms of CPU usage and storage utilization. Many organizations, even those working in the field of computer security, turn off auditing on their machines to avoid the resource penalty. Fourth, the audit trails themselves can be the target of an intruder. Intruders have been known to turn off auditing on machines in order to hide their tracks. Fifth, and last, the delay in the actual recording and analysis of the audit information can allow an intruder to do damage and exit the machine before the intrusion is noticed [17]. So, although there exists a strong desire for immediate notification of intrusive activity, audit mechanisms can introduce a delay factor.

By exploiting the broadcast property of a local area network (LAN) and network protocol standards, the analysis of network traffic can solve a number of the drawbacks associated with audit-trail-based analysis. First, network standards exist by which a variety of hosts can communicate. An intrusion detection system based on network traffic can therefore simultaneously monitor a number of hosts consisting of different hardware and operating system platforms. Second, the collection of network traffic does not create any performance degradation on the machines being monitored, so network monitoring is more attractive to a user community which places importance in the performance and responsiveness of their machines. Third, since a network monitor can be logically isolated from the computing environment, its analysis cannot be compromised by an intruder. Typically, the intruder has absolutely no way of knowing that the network is being monitored. And fourth, since a network monitor draws its information directly from the network, no delay occurs from the instant an intrusion occurs until the instant the evidence is available. Instead, intrusive activity can be observed as it occurs.

The original work on this type of network monitoring was based on simple traffic analysis: modelling the flow of data among the different machines [9,10]. In [9,10], network traffic is modelled with a concept called a data path. A data path is a method by which one machine can communicate with a second machine. A data path is defined by the three-tuple `<src_host, dst_host, network_service>`. If the traffic flow shifted (e.g., a new data path is observed) at any point, this information would be reported as a possible intrusion. For example, a particular host initiating a login to a host to which it has never logged into before would be considered suspicious. This work was based on Denning's hypothesis that intrusive activity would manifest itself as anomalous behavior [2].

Although this method showed early promise, a major drawback quickly became apparent: the information available from simple network packet analysis was at a level much too low to detect subtle intrusive activity. For example, an intrusion over a commonly used data path would not be detected. Unfortunately, this is often the case when the intrusion is being perpetrated by an insider.

To provide for a more effective intrusion detection system, our monitor needed the capability to detect

and analyze higher-level objects which are not directly observed (i.e., individual network connections and hosts). Also, to perform the analysis information about each object-attributes for the object-needed to be known.

The logical architecture of our system is shown in figure 1, and the components which provide for the additional complexity of analysis, viz. object detector and object analyzer, are shown in the dashed box. The functionality provided by these components have greatly enhanced our efforts to detect intrusive activity. Results from actual use of our monitor can be found in [7,8]. We have attempted to both generalize and formalize the methods by which our monitor detects and analyzes objects, and this work is presented in sections three and four.

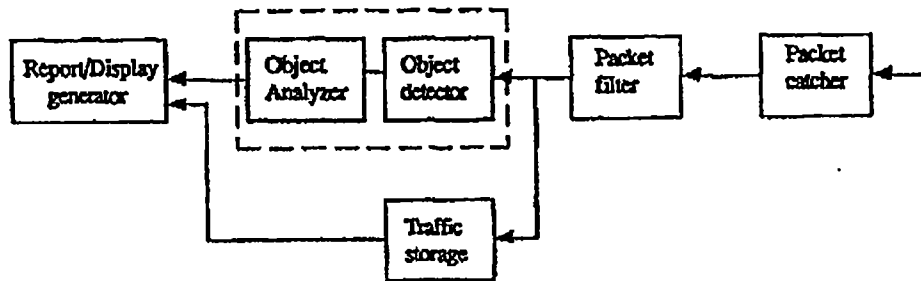


Figure 1

### 3. System Description Language

The problem of detecting intrusive activity in a heterogeneous network of computers through the observation of network packets can be generalized to the detection of a behavior in a complex system (e.g., networked system) from the analysis of low level information (e.g., network packets). The complex system is composed of a variety of components (i.e. hosts, connection, and packets) each of which in turn may be composed of other components, but only the simplest of components, the lowest levels of information, are directly visible to a monitor. Unfortunately, to detect the behavior of interest (i.e. intrusive activity), the complex components which are not directly observed, as well as the low level components, must be examined for the manifestation of the behavior.

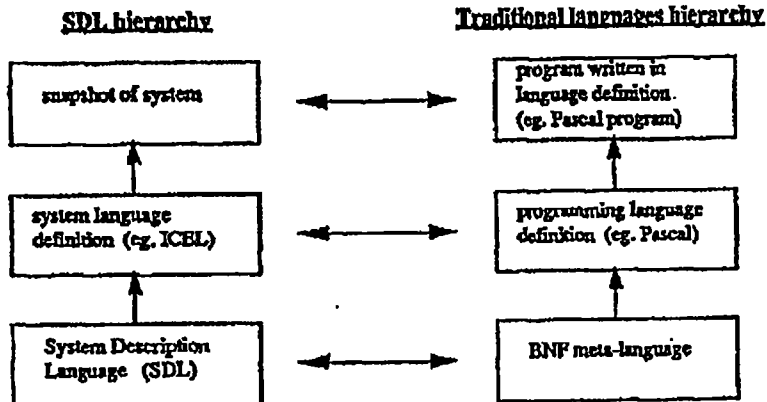


Figure 2

To provide for a formal mechanism to infer the complex components of a system, we have defined a meta-language, called the system description language (SDL), to describe the relationships among components of a system. The description of a system with this language is called its system language definition. As the

low level information is observed, the system language definition is used to infer the existence of the complex objects and the relationships between them. A snapshot of all the low level objects and the inferred complex objects and their relationships to one another represent a model of the actual system at a particular time instant. It is this model which will be examined for the manifestation of the behavior of interest.

The SDL, the system language definition, and the snapshot of an actual system have a direct resemblance to the definition of a traditional programming language. The SDL provides a functionality similar to that of the BNF meta-language. The system language definition is similar to a traditional program language definition (e.g., Pascal). And the snapshot of a system is similar to a program defined by a traditional programming language. This relationship is shown in figure 2.

The system description language is the focus of this section. Section 3.1 introduces the issues which must be addressed by the system description language. Section 3.2 presents a review of attribute grammars, the ancestor of the system description language. And section 3.3 discusses the actual system description language.

### 3.1 Issues to be Addressed by the SDL

To design a meta-language which can be used to describe and model complex systems from the observation of low level information, a number of issues must be addressed. First, how are the low level, simple components of the system detected, and how are the attributes of each low level object determined? We have chosen to not address this issue in this paper, and it is not part of the language definition. The low level components are detected, and their associated attributes are determined by a preprocessor. This is not unlike the design of conventional programming languages which assume the presence of a lexical analyzer to detect tokens, and, if necessary, determine their attributes.

The second issue is the identification and representation of components of the system which are not observed directly. In fact, a complex object which does not have a real world counterpart may be desired. For example, our model for the computer network environment includes an object called a "service-set." The service-set object does not exist in the actual system, but its presence is helpful in analyzing other components such as network connections. The system description language must provide a mechanism for inferring the existence of these unobserved, perhaps nonexistent, objects. Furthermore, the language must provide mechanisms to determine enough information about these abstract objects so they can be analyzed for the behavior of interest.

The third issue is concerned with the transitory nature of many of the objects in a system. Systems such as a heterogeneous network have a number of components which exist for a time, and then disappear. For example, network connections are created and destroyed continuously. The system description language must be able to handle the creation and destruction of components, and the system description language must provide information to determine when a component should be created or destroyed. Thus the model of an actual system, as determined by a system language definition, can change over time.

In summary, the system description language assumes that the low level, simple components and their attributes are provided to it. From these simple components, the systems description language must provide a mechanism to infer the existence of, and the relationships between, complex objects. The system description language must provide mechanisms to determine enough information about the complex objects to analyze the objects for the presence of the behavior of interest. Finally, the system description language must provide a means both to determine when a component to the system is created or destroyed and to modify the model of the system due to the creation or destruction of a component.

### 3.2 Attribute Grammars

The system description language which satisfies the above requirements is built upon the concept of attribute grammars. A quick introduction to attribute grammars is provided below. Readers already familiar with this subject may want to skip to section 3.3.

An attribute grammar describes both the strings accepted by a language (e.g., the syntax of the language) and a method to determine the "meaning" of those strings (e.g., the semantics of the language). An attribute grammar consists of a context-free grammar, a set of attributes for each symbol in the grammar, and a set of functions defined within the scope of a production rule in the grammar to determine the values for the attributes of each symbol in that production [1]. The following example of an attribute grammar for the definition and interpretation of binary numbers<sup>2</sup> will be used to clarify the relationships between these three

<sup>2</sup> This example is taken from [12].

$N \rightarrow L.L$	$N \rightarrow L_1 L_2$	$v(N) = v(L_1) + v(L_2)/2^{l(L_2)}$
$N \rightarrow L$	$N \rightarrow L$	$v(N) = v(L)$
$L \rightarrow LB$	$L_1 \rightarrow L_2 B$	$v(L_1) = 2v(L_2) + v(B), l(L_1) = l(L_2) + 1$
$L \rightarrow B$	$L \rightarrow B$	$v(L) = v(B), l(L) = 1$
$B \rightarrow 1$	$B \rightarrow 1$	$v(B) = 1$
$B \rightarrow 0$	$B \rightarrow 0$	$v(B) = 0$

A

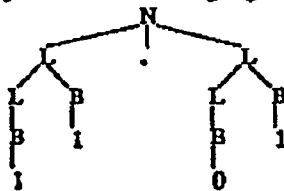
B

Figure 3

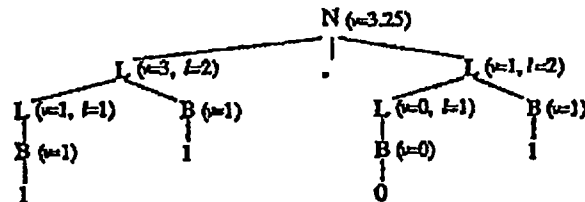
components of an attribute grammar.

The context-free grammar for our language of binary numerals is defined by  $G = (V, N, P, S)$  where  $V$  is the set of symbols,  $N$  is the set of nonterminal symbols,  $P$  is the set of production rules, and  $S$ , an element of  $N$ , is the start symbol. The set of terminal symbols, a subset of  $V$ , is  $\{1, 0, .\}$ . These are the ASCII characters one, zero, and period. The set of nonterminal symbols,  $N$ , is  $\{B, L, N\}$ . They represent the abstract objects bit, list of bits, and number. The start symbol for our attribute grammar for binary numbers is  $N$ , the abstract number. The set of production rules relating these symbols and providing the definition of acceptable strings is given in figure 3A.

By this context free grammar, we can see that the string 11.01 is an acceptable binary number. The parse tree for this string is given in figure 4A.



A



B

Figure 4

The context-free grammar can be used to build a parse tree of a string and determine whether the string is valid in the language; however, the context-free grammar cannot be used to determine the meaning of the string. The addition of attributes and attribute functions are necessary to determine the meaning of the string.

The set of attributes,  $A$ , for each nonterminal are given as follows:  $A(B) = \{v\}$ ,  $A(L) = \{v, l\}$ , and  $A(N) = \{v\}$ . The attribute  $v$  is the value of a symbol, and the attribute  $l$  is the length of a symbol.

The set of functions defined within the scope of each production rule is given in figure 3B.

By using the attributes for each symbol and the attribute functions, we can now assign meaning to each symbol in the parse tree (see figure 4B). For our language of binary numbers, the most important meaning is that of the start symbol  $N$ . Our string 11.01 now has the meaning of 3.25.

### 3.3 System Description Language

This section introduces the system description language, an extension of attribute grammars. This system description language provides a structure by which a system's components and relationships between components can be described. The description, or system language definition, of a system can be used to both infer the existence of complex objects (e.g., determine the syntactic structure of the system) and assign "meaning" to these objects (e.g., the semantic information about the system). The meaning of an object, the values of its attributes, will be used to determine if the behavior of interest is present in any of the components of the system.

Similar to an attribute grammar, a system language definition written in the SDL consists of a structural grammar, a set of attributes for each object, or symbol, in the structural grammar, and a set of

functions defined within the scope of a production rule of the structural grammar which determine the attribute values for each object in that production.

### **3.3.1 Objects**

Objects are the components of the system which will be modelled. These objects may or may not have real world counterparts. Two varieties of objects exist: basic objects and complex objects. Basic objects are the low-level components which are directly observed. These are similar to terminal symbols in traditional programming languages. Complex objects, on the other hand, are not observed and must be inferred from the observation of basic objects. Complex objects are similar to non-terminal symbols in traditional programming languages. These two objects are discussed further below.

#### **3.3.1.1 Basic Objects**

Basic objects are simple, indivisible components of the complex system being modelled; they are detected and their attributes determined by a preprocessor. This preprocessor performs the job of a lexical analyzer in traditional programming languages. Basic objects are treated as events; they only exist for the moment at which they are observed. For example, in the networked system, packets are basic objects. Basic objects for other systems may be an audit record from an operating system, a message to a spacecraft component, or a sampled data point from some measuring instrument.

A basic object type is defined by a name and a list of attributes. The name format for our system is the same as the standard C identifier. Attributes will be discussed in section 3.3.2. An example basic object representing a possible network packet is:

basic: packet ( *attribute list* )

The keyword *basic* states that the following object type is a basic object, and the object type's name is *packet*. Attributes for this object will be discussed later in section 3.3.3.

#### **3.3.1.2 Complex Objects**

As mentioned previously, complex objects are components of a system which are not directly observed by the monitor, so they must be inferred from the observation of the basic objects. A complex object is composed of basic objects and/or other complex objects. For example, a complex object type called *process* may be defined for an audit-trail-based monitor. Although processes are not directly observed by the monitor, information about them can be inferred from the audit records. Therefore, in our model, processes are composed of audit records. Compositions will be discussed further in section 3.3.3.

A major difference between complex objects and basic objects is that complex objects have persistence. Whereas basic objects are treated as events, complex objects are treated as persistent elements which are created and possibly destroyed. The creation of a complex object occurs as soon as it can be inferred. The destruction of an object is considerably more difficult and depends on both the definition of the complex object and the existence of objects which compose the complex object. The two rules which govern the possible destruction of an object are described below.

First, if any object A exists and is part of an object B's composition, then object B should continue to exist. Second, if the last object which is part of object B's composition is destroyed, then object B will be destroyed after a specified time delay,  $\Delta t$ , unless another object which is part of B's composition is created or observed. This specified  $\Delta t$  is the value of a function associated with the object, and it may depend on the object's attributes.

Complex objects can be composed of only basic objects, only complex objects, or a combination of basic and complex objects. Complex object types are defined in my system by one of the following forms depending on their composition:

complex type *i*:      *name* ( *attribute list* )

where *i* varies from 1 to 3 depending on the makeup of the composition objects.

### **3.3.2 Attributes**

As mentioned previously, each object has a set of attributes associated with it. These attributes provide a "meaning" to each object. It is the attributes which will be used to determine if this object is associated with a particular behavior. These attributes are also used, along with the production rules described in section 3.3.3, to determine if an object A is part of object B's composition.

Each attribute consists of a name and a type. The name is used to reference the value, and the type determines the value type which can be assigned or retrieved from the attribute. For example, "int value" would

describe an attribute of type "int" which is referenced by the name "value." Attribute types may be complex structures defined in the same format as complex types are described in the C language [11].

Many of the attribute values of an object will be assigned by the monitor. For example, when the existence of a new host is inferred, a host object is created and its internet address is immediately assigned by the monitor. The values of other attributes, however, are determined by attribute functions. Attribute functions, described in section 3.3.4, take as input attribute values associated with the object and possibly attribute values of other objects associated with it by the production rules (see section 3.3.3).

A complex object type to represent a stream (a unidirectional flow of data from one process to another process) composed of packets can now be defined as follows:

```
complex type 1:      stream{
                        inet_addr      src_addr
                        inet_addr      dst_addr
                        int             src_port
                        int             dst_port
                        int             creation_time
                        int             num_of_packets
                        int             num_of_bytes
                      }
```

This simple definition of a process has a simple identifier, *stream*, addresses for the source and destination hosts, source and destination ports to specify the processes on the two machines, a time of creation, the number of packets exchanged between the two processes, and the number of bytes in all the packets exchanged.

The set of attributes for an object *O* can be defined as  $A(O) = \{a_1, a_2, \dots, a_n\}$ . For example,  $A(\text{process}) = \{\text{src\_addr}, \text{dst\_addr}, \text{src\_port}, \text{dst\_port}, \text{creation\_time}, \text{num\_of\_packets}, \text{num\_of\_bytes}\}$ .

### 3.3.3 Productions

Productions define the relationship between the different object types of a system. They define which types of objects compose a complex object, and they indicate how to determine which set of objects from an object type compose the complex object. A production rule has the form:

*complex\_object\_type*  $\rightarrow$  list of *object\_composition*

The *complex\_object\_type* is simply the name of a complex object type (e.g., *stream*). An *object\_composition* is a set defined by a tuple of the form  $\langle \text{object\_type}, \text{restrictions} \rangle$ . The *object\_type* is simply the name of one of the defined object types (basic or complex), and the restrictions determine which of all possible objects of type *object\_type* are actually used to compose the complex object.

For example, let the complex object type called *stream* be defined as above, and let the object type called *packet* be defined as follows:

```
basic:      packet {
                        inet_addr      src_addr
                        inet_addr      dst_addr
                        int             src_port
                        int             dst_port
                        int             num_of_bytes
                        int             time
                      }
```

A production rule for the *stream* object can now be defined as follows:

```
stream  $\rightarrow$  packet
where for all  $e \in \text{packet}$ 
  e.src_addr = stream.src_addr
  & e.dst_addr = stream.dst_addr
  & e.src_port = stream.src_port
  & e.dst_port = stream.dst_port
```

Finally, each element of *packet* which composes a particular *stream* object is called a sub-component of the *stream* object, and the *stream* object is called a super-component of the *packet* objects. The concepts of sub-components and super-components will be used in section 4.2 to define integrated object analysis functions.

### 3.3.4. Attribute Functions

The attributes of a complex object which are not defined by the monitor when the object is inferred are defined by attribute functions. The attribute functions for a structural language are defined as they are for attribute grammars; however, special attention must be given to the format of the production and the restriction for the production. For example, an attribute function to determine the value for the attribute "num\_of\_bytes" of a stream object could be as follows:

$$\text{stream.num\_of\_bytes} = \sum_{i=1}^n c_i.\text{num\_of\_bytes}$$

Where  $n = |\text{packet}|$ , and each  $c \in \text{packet}$  is assumed to be a sub-component of the stream object as defined by the restrictions in the production rule for stream objects.

## 4. Detecting Behaviors in Systems

Once the structural grammar, attributes, and attribute functions have been defined, a second set of functions, called behavior-detection functions, must be defined for each object in the structural grammar. Behavior-detection functions determine whether an object is associated with the particular behavior of interest. Because a behavior may manifest itself differently or more clearly in different object types, each object in a system parse tree (the snapshot of the system) must be examined for the behavior by particular behavior-detection functions designed for that object type. For each type of object, there will be two behavior-detection functions: the isolated behavior-detection function and the integrated behavior-detection functions. These two function types are discussed below.

### 4.1 Isolated Object Analysis

An isolated behavior-detection function for an object uses the attributes of that object to calculate the probability that the object is associated with the behavior of interest. In short, an isolated behavior-detection function is a classifier. With some preprocessing to transform the attribute types, a large number of classifiers can be used.

Unfortunately, classifiers generally have to be trained with sample data, and the behavior of interest is often quite rare. There are at least two possible solutions to the problem of lack of sample data: expert systems and single behavior classifiers. An expert system, designed by people knowledgeable about the problem domain, can use heuristics to determine how close an object's behavior is to the behavior of interest. A single behavior classifier is built around the assumption that a rare behavior will be significantly different than normal behavior [2]. If this is true, a single classifier can profile normal behavior, and then it could report any behavior which does not strongly resemble normal behavior. Work on such single behavior classifiers have been performed by SRI for IDES [13] and Los Alamos National Laboratory for Wisdom and Sense [17]. For our particular problem environment, we combined the efforts of both an expert system and a single behavior classifier.

### 4.2 Integrated Objects Analysis

An integrated behavior-detection function for an object modifies the result of the isolated behavior-detection function for the object by including the analysis of the isolated behavior-detection functions for sub-components and super-components of that object. The modification by an integrated behavior-detection function allows the inclusion of both the results of aggregated analysis (those from super-components) and the results of more detailed levels of analysis (those from sub-components). The integrated behavior-detection function can be implemented by a weighted average function such as:

$$\frac{W_1 * \text{Object\_if} + W_2 * \text{Super\_if} + W_3 * \text{Sub\_if}}{W_1 + W_2 + W_3}$$

Where Object\_if is the value calculated by the object's isolated behavior-detection function, Super\_if is the average isolated behavior-detection function value for all the super-components, Sub\_if is the average isolated behavior-detection function value for all the sub-components, and  $W_1$ ,  $W_2$ , and  $W_3$  are the weights.

The relationship between an object's attributes, isolated behavior-detection functions, and integrated behavior-detection functions can be seen in figure 5. In this example, we are interested in analyzing the object  $B_1$  for a particular behavior. The object  $B_1$  is composed of objects  $C_1$  and  $C_2$ , and it is part of the object  $A_1$ . Result  $B_{1v}$  is the analysis of object  $B_1$  in isolation, and result  $B_{1v'}$  is the result after combining the result of  $B_{1v}$  with the results from objects  $C_1$ ,  $C_2$ , and  $A_1$ .

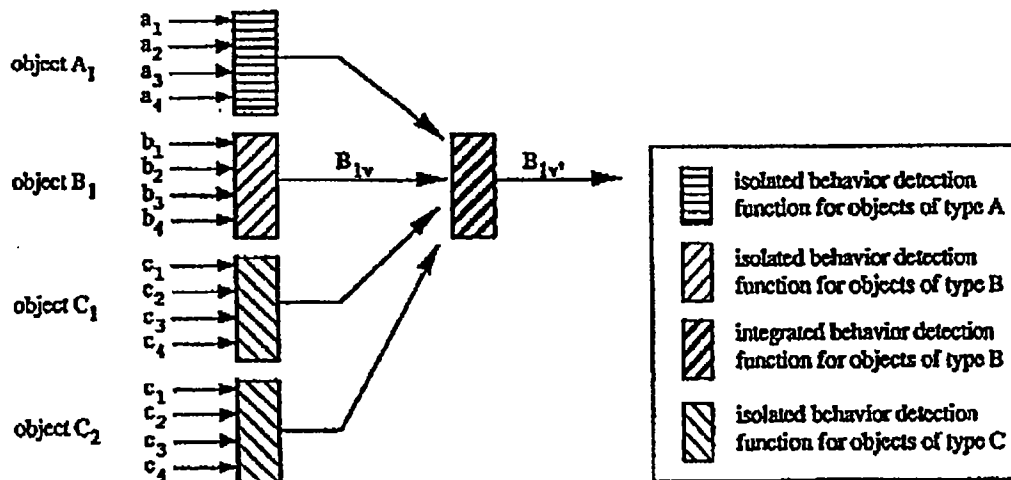


Figure 5

### 5. Results and Future Research

By using the system language definition for the networked environment described in [7], one programmer was able to code both the object detector and object analyzer modules in less than two weeks. Since the coding of these modules is a straight forward implementation of the system language definition, we hope to provide automatic development tools in the future which will automatically generate object detector and object analyzer modules from a system language definition.

We have concentrated our analysis efforts on an isolated behavior-detection function for connections. This function combines a simple anomaly detector, an attack model, and an expert system to arrive at a single suspicion value. The higher the suspicion value is, the more likely our monitor believes the connection is associated with intrusive activity.

We monitored the Electrical Engineering and Computer Science LAN at UCD for a period of approximately three months. During this time over 400,000 connections were detected and analyzed, and among these connections, over 400 were identified as being associated with intrusive behavior.

Our future work includes continual improvement of the isolated behavior-detection function for connections as well as other objects in the model (i.e. service-sets, hosts, and streams). We would like to take advantage of semantic knowledge about known system vulnerabilities, and we would also like to develop profiles of intrusive activity as well as normal activity.

As mentioned previously, we are also moving towards automatic code generation for the object detector and object analyzer components of the monitor. We are currently developing a system language definition for a stand alone host based monitor too, and if we can develop automatic code generators for object detector and analyzer modules, then porting the monitor to a different operating system should be greatly simplified.

Finally, we are incorporating our network monitor into a distributed intrusion detection system called DIDS [15]. DIDS combines both host based as well as network based monitors to take advantage of the benefits of both systems.

### References

1. G.V. Bochmann, "Semantic Evaluation from Left to Right," *Communications of the ACM*, vol. 19, no. 2, pp. 55-62, Feb. 1976.
2. D.E. Denning, "An Intrusion Detection Model," *IEEE Trans. on Software Engineering*, vol. SE-13, no. 2, pp. 222-232, Feb. 1987.
3. P.J. Denning, ed. *Computers Under Attack: Intruders, Worms, and Viruses*. New York: ACM Press, 1990.
4. Department of Defense Trusted Computer System Evaluation Criteria, Dept. of Defense, National Computer Security Center, DOD 5200.28-STD, Dec. 1985.
5. G.V. Dias, K.N. Levitt, B. Mukherjee, "Modeling Attacks on Computer Systems: Evaluating Vulnerabilities and Forming a Basis for Attack Detection," Technical Report CSE-90-41, University of California, Davis.
6. C. Dowell and P. Ramstedt, "The COMPUTERWATCH Data Reduction Tool," *Proc. 13th National Computer Security Conference*, pp. 99-108, Washington, D.C., Oct 1990.
7. L.T. Heberlein, "Towards Detecting Intrusions in a networked Environment," Technical Report CSE-91-23, University of California, Davis.
8. L.T. Heberlein, B. Mukherjee, K.N. Levitt, D. Mansur, "Towards Detecting Intrusions in a Networked Environment," *Proc. 14th Department of Energy Computer Security Group Conference*, May 1991.
9. L.T. Heberlein, G.V. Dias, K.N. Levitt, B. Mukherjee, J. Wood, "Network Attacks and an Ethernet-based Network Security Monitor," *Proc. 13th Department of Energy Computer Security Group Conference*, pp. 14.1-14.13, May 1990.
10. L.T. Heberlein, G.V. Dias, K. N. Levitt, B. Mukherjee, J. Wood, D. Wolber, "A Network Security Monitor," *Proc. 1990 Symposium on Research in Security and Privacy*, pp. 296-304, May 1990.
11. B.W. Kernigan, D.M. Ritchie, *The C Programming Language*, 2nd ed. Englewood Cliffs, New Jersey: Prentice Hall, 1988.
12. D.E. Knuth, "Semantics of Context-Free Languages," *Math Systems Th.* 2 (1968), 127-145. Correction appears in *Math Systems Th.* 5 (1971), 95.
13. T.P. Lunt, et al., "A Real Time Intrusion Detection Expert System (IDES)," Interim Progress Report, Project 6784, SRI International, May 1990.
14. S.R. Smaha, "Haystack: An Intrusion Detection System," *Proc. IEEE Fourth Aerospace Computer Security Applications Conference*, Orlando, FL, Dec. 1988.
15. S.R. Snapp, J. Brentano, G.V. Dias, T.L. Goan, L.T. Heberlein, C. Ho, K.N. Levitt, B. Mukherjee, S.R. Smaha, T. Grance, D.M. Teal, D.L. Mansur, "DIDS (Distributed Intrusion Detection System) - Motivation, Architecture, and an Early Prototype," to be published in *Proc. 14th National Computer Security Conference*, Oct. 1991.
16. W.T. Tener, "Discovery: an expert system in the commercial data security environment," *Security and Protection in Informations Systems: Proc. Fourth IFIO TC11 International Conference on Computer Security*, North-Holland, May 1988.
17. H.S. Vaccaro and G.E. Liepins, "Detection of Anomalous Computer Session Activity," *Proc. 1990 Symposium on Research in Security and Privacy*, pp. 280-289, Oakland, CA, May 1989.
18. J.R. Winkler, "A Unix Prototype for Intrusion and Anomaly detection in Secure Networks," *Proc. 13th National Computer Security Conference*, pp. 115-124, Washington, D.C., Oct. 1990.

*NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY/  
NATIONAL COMPUTER SECURITY CENTER*

# **14TH NATIONAL COMPUTER SECURITY CONFERENCE**

**October 1-4, 1991  
Omni Shoreham Hotel  
Washington, D.C.**



SYM\_P\_0069365

**EXHIBIT K**

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

**EMERALD 1997 invalidates the indicated claims under 35 U.S.C. § 102(b) and 35 U.S.C. § 103\***

All text citations are taken from: P. Porras and P. Neumann, "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances", 20<sup>th</sup> NISSC October 9, 1997 [SYM\_P\_0535485- SYM\_P\_0535497] ("EMERALD 1997").

SRI admits this paper was published on Oct. 9, 1997 in the Proceedings of the 20<sup>th</sup> National Information Systems Security Conference. See SRI Response to ISS's First Set of RFA's, #1.

The text included herein are merely representative samples of the disclosure in the asserted reference.

\* 35 U.S.C. § 103 reference citations are identified under the heading "103" and are taken from: L.T. Heberlein et al., "A Method to Detect Intrusive Activity in a Networked Environment," 14<sup>th</sup> National Computer Security Conference, Oct. 1-4, 1991 [SYM\_P\_0069355- SYM\_P\_0069365] ("Intrusive Activity 1991").

# **EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances** **"EMERALD 1997"**

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
1	A computer-automated method of hierarchical event monitoring and analysis within an enterprise network comprising:	<p><i>"The EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances) environment is a distributed scalable tool suite for tracking malicious activity through and across large networks. EMERALD introduces a highly distributed, building-block approach to network surveillance, attack isolation, and automated response. It combines models from research in distributed high-volume event-correlation methodologies with over a decade of intrusion detection research and engineering experience. The approach is novel in its use of highly distributed, independently tunable, surveillance and response monitors that are deployable polymorphically at various abstract layers in a large network. These monitors contribute to a streamlined event-analysis system that combines signature analysis with statistical profiling to provide localized real-time protection of the most widely used network services on the Internet."</i></p> <p>p. 353 [SYM_P_0535485]</p> <p><i>"The typical target environment of the EMERALD project is a large enterprise network with thousands of users connected in a federation of independent administrative domains. Each administrative domain is viewed as a collection of local and network services that provide an interface for requests from individuals internal and external to the domain."</i></p> <p>p. 354 [SYM_P_0535486]</p> <p><i>"EMERALD introduces a hierarchically layered approach to network surveillance that includes service analysis covering the misuse of individual components and network services within the boundary of a single domain; domain-wide analysis covering misuse visible across multiple services and components; and enterprise-wide analysis covering coordinated misuse across multiple domains."</i></p> <p>p. 355 [SYM_P_0535487]</p> <p><i>"The typical target environment of the EMERALD project is a large enterprise network with thousands of users connected in a federation of independent administrative domains. Each administrative domain is viewed as a collection of local and network services that provide an interface for requests from individuals internal and external to the domain."</i></p> <p>p. 354 [SYM_P_0535486]</p> <p><i>"We introduce the concept of dynamically deployable, highly distributed, and independently tunable service monitors. Service</i></p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'203 Claim number	Claim Term	<p style="text-align: center;"><b>EMERALD 1997</b> (printed publication)</p> <p>monitors are dynamically deployed within a domain to provide localized real-time analysis of infrastructure (e.g., routers or gateways) and services (privileged subsystems with network interfaces). Service monitors may interact with their environment passively (reading activity logs) or actively via probing to supplement normal event gathering. This localized coverage of network services and domain infrastructure forms the lowest tier in EMERALD's layered network-monitoring scheme." p. 355 [SYM_P_0535487]</p> <p>"All EMERALD monitors (service, domain, and enterprise) are implemented using the same monitor code-base." p. 357 [SYM_P_0535489]</p> <p>"EMERALD employs a building-block architectural strategy using independent distributed surveillance monitors that can analyze and respond to malicious activity on local targets, and can interoperate to form an analysis hierarchy. This layered analysis hierarchy provides a framework for the recognition of more global threats to interdomain connectivity, including coordinated attempts to infiltrate or destroy connectivity across an entire network enterprise." p. 355 [SYM_P_0535487]</p> <p>"In general, a monitor may include additional analysis engines that may implement other forms of event analysis, or a monitor may consist of only a single resolver implementing a response policy based on intrusion summaries produced by other EMERALD monitors."</p> <p>Multiple analysis engines implementing different analysis methods may be employed to analyze a variety of event streams that pertain to the same analysis target. These analysis engines are intended to develop significantly lower volumes of abstract <i>intrusion</i> or <i>suspicion reports</i>. The profiler and signature engines receive large volumes of event logs specific to the analysis target, and produce smaller volumes of intrusion or suspicion reports that are then fed to their associated resolver." p. 356 [SYM_P_0535488]</p>
	detecting, by the network monitors, suspicious network activity	
	based on analysis of	

# EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances "EMERALD 1997"

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
	<p>network traffic data selected from the following categories: {network packet data transfer commands, network packet data transfer errors, network packet data volume, network connection requests, network connection denials, error codes included in a network packet};</p>	<p>"Event Generation and Storage: Audit generation and storage has tended to be a centralized activity, and often gathers excessive amounts of information at inappropriate layers of abstraction. Centralized audit mechanisms place a heavy burden on the CPU and I/O throughput, and simply do not scale well with large user populations. In addition, it is difficult to extend centralized audit mechanisms to cover spatially distributed components such as network infrastructure (e.g., routers, filters, DNS, firewalls) or various common network services."</p> <p>p. 354 [SYM_P_0535486]</p> <p>"Service monitors are dynamically deployed within a domain to provide localized real-time analysis of infrastructure (e.g., routers or gateways) and services (privileged subsystems with network interfaces). Service monitors may interact with their environment passively (reading activity logs) or actively via probing to supplement normal event gathering."</p> <p>p. 355 [SYM_P_0535487]</p> <p>"The generic EMERALD monitor architecture is illustrated in Figure 1. The architecture is designed to enable the flexible introduction and deletion of analysis engines from the monitor boundary as necessary. In its dual-analysis configuration, an EMERALD monitor instantiation combines signature analysis with statistical profiling to provide complementary forms of analysis over the operation of network services and infrastructure. In general, a monitor may include additional analysis engines that may implement other forms of event analysis, or a monitor may consist of only a single resolver implementing a response policy based on intrusion summaries produced by other EMERALD monitors. Monitors also incorporate a versatile application programmers' interface that enhances their ability to interoperate with the analysis target, and with other third-party intrusion-detection tools.</p> <p>Underlying the deployment of an EMERALD monitor is the selection of a target-specific event stream. The event stream may be derived from a variety of sources including audit data, network datagrams, SNMP traffic, application logs, and analysis results from other intrusion-detection instrumentation. The event stream is parsed, filtered, and formatted by the target-specific event-collection methods provided within the resource object definition (see Section III-B). Event records are then forwarded to the monitor's analysis engine(s) for processing.</p> <p>EMERALD's <i>profiler engine</i> performs statistical profile-based anomaly detection given a generalized event stream of an analysis target (Section III-C). EMERALD's <i>signature engine</i> requires minimal state-management and employs a rule-coding scheme that breaks</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'203 Claim number	Claim Term	<p style="text-align: center;"><b>EMERALD 1997</b> (printed publication)</p> <p>from traditional expert-system techniques to provide a more focused and distributed signature-analysis model (Section III-D). Multiple analysis engines implementing different analysis methods may be employed to analyze a variety of event streams that pertain to the same analysis target." p. 356 [SYM_P_0535488]</p> <p>"Similarly, the analysis engines are responsible for establishing and maintaining a communication link with a target event collection method (or event filter) and prompting the reconfiguration of the collection method's filtering semantics when necessary. Event collection methods provide analysis engines with target-specific event records upon which the statistical and signature analyses are performed." p. 362 [SYM_P_0535494]</p> <p>"EMERALD also provides a framework for recognizing more global threats to interdomain connectivity, including coordinated attempts to infiltrate or destroy connectivity across an entire network enterprise. ... It generalizes to network environments the Safeguard experience [2], which overcame profile explosion and scalability problems by locally profiling the activities of subsystems and commands rather than of individual users." p. 364 [SYM_P_0535496]</p> <p>"EMERALD also extends the statistical-profile model of NIDES, to analyze the operation of network services, network infrastructure, and activity reports from other EMERALD monitors. Various other efforts have considered one of the two types of analysis - signature-based (e.g., Porras [18] has used a state-transition approach; the U.C. Davis and Trident DIDS [4] addresses abstracted analysis for networking, but not scalability; the Network Security Monitor [7] seeks to analyze packet data rather than conventional audit trails; Purdue [5] seeks to use adaptive-agent technology) or profile-based. More recent work in UC Davis' GridS effort [24] employs <i>activity graphs</i> of network operations to search for traffic patterns that may indicate network-wide coordinated attacks." p. 364 [SYM_P_0535496]</p> <p><i>See also Declaration of Frederick Avolio</i></p>
-------------------------	------------	---

# EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances "EMERALD 1997"

203 Claim number	Claim Term	EMERALD 1997 (printed publication)
		<p>103:</p> <p>"For example, in the networked system, packets are basic objects. ... An example basic object representing a possible network packet is: basic: packet { attribute list }"</p> <p><i>Intrusive Activity 1991</i> at 367 [SYM_P_0069360]</p> <p>"This simple definition of a process has a simple identifier, <i>stream</i>, addresses for the source and destination hosts, source and destination ports to specify the processes on the two machines, a time of creation, the number of packets exchanged between the two processes, and the number of bytes in all the packets exchanged."</p> <p><i>Intrusive Activity 1991</i> at 368 [SYM_P_0069361]</p> <p>"For example, an attribute function to determine the value for the attribute "num_of_bytes" of a stream object could be as follows:</p> <p><math>stream\_num\_of\_bytes = \sum_{i=1}^n c_i * num\_of\_bytes</math>"</p> <p><i>Intrusive Activity 1991</i> at 369 [SYM_P_0069362]</p> <p>"network connections are created and destroyed continuously"</p> <p><i>Intrusive Activity 1991</i> at 365 [SYM_P_0069358]</p> <p>"We have concentrated our analysis efforts on isolated behavior-detection functions for connections. ... The higher the suspicious value is, the more likely our monitor believes the connection is associated with intrusive activity. We monitored the Electrical Engineering and Computer Science LAN at UCD for a period of approximately three months. During this time over 400,000 connections were detected and analyzed, and among these connections, over 400 were identified as being associated with intrusive behavior."</p> <p><i>Intrusive Activity 1991</i> at 370 [SYM_P_0069363]</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**“EMERALD 1997”**

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
	generating, by the monitors, reports of said suspicious activity; and	<p>“Information correlated by a service monitor can be disseminated to other EMERALD monitors through a <i>subscription</i>-based communication scheme. Subscription provides EMERALD's message system both a push and pull data exchange capability between monitor interoperation (see Section III-F). EMERALD client monitors are able to subscribe to receive the analysis results that are produced by server monitors. As a monitor produces analysis results, it is then able to disseminate these results asynchronously to its client subscribers. Through subscription, EMERALD monitors distributed throughout a large network are able to efficiently disseminate reports of malicious activity without requiring the overhead of synchronous polling.”  pp. 355-56 [SYM_P_0535487- SYM_P_0535488]</p> <p>“Multiple analysis engines implementing different analysis methods may be employed to analyze a variety of event streams that pertain to the same analysis target. These analysis engines are intended to develop significantly lower volumes of abstract <i>intrusion</i> or <i>suspicion reports</i>. The profiler and signature engines receive large volumes of event logs specific to the analysis target, and produce smaller volumes of intrusion or suspicion reports that are then fed to their associated resolver.”  p. 356 [SYM_P_0535488]</p> <p>“Event records are defined based on the contents of the monitor's target event stream(s). Analysis result structures are used to package the findings produced by the analysis engine. Event records and analysis results are defined similarly to allow the eventual hierarchical processing of analysis results as event records by subscriber monitors.”  p. 358 [SYM_P_0535490]</p> <p>“Externally, EMERALD monitors interoperate with one another in a manner analogous to internal communication: service monitors produce local analysis results that are passed to the domain monitor; domain monitors correlate service monitor results, producing new results that are further propagated to enterprise monitors; enterprise monitors correlate and respond to the analysis results produced by domain monitors. ... Through the internal message system, the resolver submits configuration requests and probes to the analysis engines, and receives from the analysis engines their analysis results. The analysis engines operate as servers providing the resolver with intrusion or suspicion reports either asynchronously or upon request.”  p. 362 [SYM_P_0535494]</p>

# EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances "EMERALD 1997"

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
	<p>automatically receiving and integrating the reports of suspicious activity, by one or more hierarchical monitors.</p>	<p>"Domain-wide analysis forms the second tier of EMERALD's layered network surveillance scheme. A <i>domain monitor</i> is responsible for surveillance over all or part of the domain. <i>Domain monitors</i> correlate intrusion reports disseminated by individual service monitors, providing a domain-wide perspective of malicious activity (or patterns of activity). In addition to domain surveillance, the domain monitor is responsible for reconfiguring system parameters, interfacing with other monitors beyond the domain, and reporting threats against the domain to administrators. Lastly, EMERALD enables enterprise-wide analysis, providing a global abstraction of the cooperative community of domains. Enterprise-layer monitors correlate activity reports produced across the set of monitored domains. Enterprise-layer monitors focus on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, and coordinated attacks from multiple domains against a single domain. Through this correlation and sharing of analysis results, reports of problems found by one monitor may propagate to other monitors throughout the network."</p> <p>p. 356 [SYM_P_0535488]</p> <p>"Decision Unit Configuration: This refers to the semantics used by the resolver's decision unit for merging the analysis results from the various analysis engines. The semantics include the response criteria used by the decision unit for invoking countermeasure handlers."</p> <p>p. 358 [SYM_P_0535490]</p> <p>"Implementation of the response policy, including coordinating the dissemination of the analysis results, is the responsibility of the EMERALD resolver. The resolver is an expert system that receives the intrusion and suspicion reports produced by the profiler and signature engines, and based on these reports invokes the various response handlers defined within the resource object."</p> <p>... EMERALD supports extensive intermonitor sharing of analysis results throughout its layered analysis architecture. Resolvers are able to request and receive intrusion reports from other resolvers at lower layers in the analysis hierarchy. As analysis results are received from subscribers, they are forwarded via the monitor's event filters to the analysis engines. This tiered collection and correlation of analysis results allows EMERALD monitors to represent and profile more global malicious or anomalous activity that is not visible from the local monitoring of individual network services and assets (see Section IV).</p>

# EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances "EMERALD 1997"

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
		<p>... The resolver operates as the center of intramonitor communication. As the analysis engines build intrusion and suspicion reports, they propagate these reports to the resolver for further correlation, response, and dissemination to other EMERALD monitors. ... For example, an intrusion report produced by a service monitor in one domain could be propagated to an enterprise monitor, which in turn sensitizes service monitors in other domains to the same activity." p. 360-61 [SYM_P_0535492- SYM_P_0535493]</p> <p>"EMERALD monitors incorporate a duplex messaging system that allows them to correlate activity summaries and countermeasure information in a distributed hierarchical analysis framework. ... Once the subscription request is accepted by the server, the server module forwards events or analysis results to the client automatically as data becomes available, and may dynamically reconfigure itself as requested by the client's control requests."</p> <p>... Intermonitor communication also operates using the subscription-based hierarchy. A domain monitor subscribes to the analysis results produced by service monitors, and then propagates its own analytical results to its parent enterprise monitor. The enterprise monitor operates as a client to one or more domain monitors, allowing them to correlate and model enterprise-wide activity from the domain-layer results. Domain monitors operate as servers to the enterprise monitors, and as clients to the service-layer monitors deployed throughout their local domain. This message scheme would operate identically if correlation were to continue at higher layers of abstraction beyond enterprise analysis." p. 361-62 [SYM_P_0535493- SYM_P_0535494]</p> <p>"Through the internal message system, the resolver submits configuration requests and probes to the analysis engines, and receives from the analysis engines their analysis results. The analysis engines operate as servers providing the resolver with intrusion or suspicion reports either asynchronously or upon request." p. 362 [SYM_P_0535494]</p> <p>"Domain monitors may also operate within an enterprise hierarchy, where they disseminate intrusion reports to enterprise monitors for</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**"EMERALD 1997"**

'203 Claim number	Claim Term	<p style="text-align: center;"><b>EMERALD 1997</b> (printed publication)</p>
		<p>global correlation. Where trust exists between domains, peer-to-peer subscription provides a useful technique for keeping domains sensitized to malicious activity occurring outside their view. Enterprise-layer monitors attempt to model and detect coordinated efforts to infiltrate domain perimeters or prevent interconnectivity between domains. Enterprise surveillance may be used where domains are interconnected under the control of a single organization, such as a large privately owned WAN. Enterprise surveillance is very similar to domain surveillance: the <i>enterprise monitor</i> subscribes to various domain monitors, just as the domain monitors subscribed to various local service monitors. The enterprise monitor (or monitors, as it would be important to avoid centralizing any analysis) focuses on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, or coordinated attacks from multiple domains against a single domain."</p> <p>p. 363 [SYM_P_0535495]</p>
2	<p>The method of claim 1, wherein integrating comprises correlating intrusion reports reflecting underlying commonalities.</p>	<p>"Domain monitors correlate intrusion reports disseminated by individual service monitors, providing a domain-wide perspective of malicious activity (or patterns of activity)."</p> <p>p. 356 [SYM_P_0535488]</p> <p>"Above the service layer, signature engines scan the aggregate of intrusion reports from service monitors in an attempt to detect more global coordinated attack scenarios or scenarios that exploit interdependencies among network services."</p> <p>p. 360 [SYM_P_0535492]</p> <p>"Domain monitors may also operate within an enterprise hierarchy, where they disseminate intrusion reports to enterprise monitors for global correlation. Where trust exists between domains, peer-to-peer subscription provides a useful technique for keeping domains sensitized to malicious activity occurring outside their view. Enterprise-layer monitors attempt to model and detect coordinated efforts to infiltrate domain perimeters or prevent interconnectivity between domains."</p> <p>...</p> <p>The enterprise monitor (or monitors, as it would be important to avoid centralizing any analysis) focuses on network-wide threats such</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'203 Claim number	Claim Term	<p style="text-align: center;"><b>EMERALD 1997</b> (printed publication)</p> <p>as Internet worm-like attacks, attacks repeated against common network services across domains, or coordinated attacks from multiple domains against a single domain. As an enterprise monitor recognizes commonalities in intrusion reports across domains (e.g., the spreading of a worm or a mail system attack repeated throughout the enterprise), its resolver can take steps to help domains counter the attack, and can also help sensitize other domains to such attacks before they are affected." p. 363 [SYM_P_0535495]</p>
3	The method of claim 1, wherein integrating further comprises invoking countermeasures to a suspected attack.	<p>"In addition to domain surveillance, the domain monitor is responsible for reconfiguring system parameters, interfacing with other monitors beyond the domain, and reporting threats against the domain to administrators." p. 356 [SYM_P_0535488]</p> <p>[See Fig. 1 label: RESOLVER (Countermeasure Unit)]</p> <p>"• <b>Decision Unit Configuration:</b> This refers to the semantics used by the resolver's decision unit for merging the analysis results from the various analysis engines. The semantics include the response criteria used by the decision unit for invoking countermeasure handlers....</p> <p>• <b>Valid Response Methods:</b> Various response functions can be made available to the resolver as it receives intrusion reports from its analysis engines or intrusion summaries from subscribers. These are pre-programmed countermeasure methods that the resolver may invoke as intrusion summaries are received." p. 358 [SYM_P_0535490]</p> <p>"Implementation of the response policy, including coordinating the dissemination of the analysis results, is the responsibility of the EMERALD resolver. The resolver is an expert system that receives the intrusion and suspicion reports produced by the profiler and signature engines, and based on these reports invokes the various response handlers defined within the resource object."</p> <p>... In addition to its external-interface responsibilities, the resolver operates as a fully functional decision engine, capable of invoking real-time countermeasures in response to malicious or anomalous activity reports produced by the analysis engines. Countermeasures are</p>

# EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances "EMERALD 1997"

'203 Claim number	Claim Term	<p>EMERALD 1997 (printed publication)</p> <p>defined in the response-methods field of the resource object. Included with each valid response method are evaluation metrics for determining the circumstances under which the method should be dispatched. These response criteria involve two evaluation metrics: a threshold metric that corresponds to the measure values and scores produced by the profiler engine, and severity metrics correspond to subsets of the associated attack sequences defined within the resource object. The resolver combines the metrics to formulate its monitor's response policy. Aggressive responses may include direct countermeasures such as closing connections or terminating processes. More passive responses may include the dispatching of integrity-checking handlers to verify the operating state of the analysis target." p. 360-61 [SYM_P_0535492- SYM_P_0535493]</p>
4	<p>The method of claim 1, wherein the plurality of network monitors include an API for encapsulation of monitor functions and integration of third-party tools.</p>	<p>"Monitors also incorporate a versatile application programmers' interface that enhances their ability to interoperate with the analysis target, and with other third-party intrusion-detection tools." p. 356 [SYM_P_0535488]</p> <p>"Interoperability is especially critical to EMERALD's decentralized monitoring scheme, and extends within EMERALD's own architectural scope as well as to third-party modules.</p> <p>... Third-party modules may also submit and receive analysis results via the resolver's external interfaces. This will allow third-party modules to incorporate the results from EMERALD monitors into their own surveillance efforts, or to contribute their results to the EMERALD analysis hierarchy. Lastly, the monitor's internal API allows third-party analysis engines to be linked directly into the monitor boundary." p. 357 [SYM_P_0535489]</p>
5	<p>The method of claim 1, wherein the enterprise network is a TCP/IP network.</p>	<p>"The EMERALD (Event Monitoring Enabling Responses to Anomalous Live Disturbances) environment is a distributed scalable tool suite for tracking malicious activity through and across large networks. EMERALD introduces a highly distributed, building-block approach to network surveillance, attack isolation, and automated response. It combines models from research in distributed high-</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**"EMERALD 1997"**

'203 Claim number	Claim Term	<p style="text-align: center;"><b>EMERALD 1997</b> (printed publication)</p> <p><i>volume event-correlation methodologies with over a decade of intrusion detection research and engineering experience. The approach is novel in its use of highly distributed, independently tunable, surveillance and response monitors that are deployable polymorphically at various abstract layers in a large network. These monitors contribute to a streamlined event-analysis system that combines signature analysis with statistical profiling to provide localized real-time protection of the most widely used network services on the Internet.</i></p> <p>p. 353 [SYM_P_0535485]</p> <p>"The typical target environment of the EMERALD project is a large enterprise network with thousands of users connected in a federation of independent administrative domains. Each administrative domain is viewed as a collection of local and network services that provide an interface for requests from individuals internal and external to the domain. Network services include features common to many network operating systems such as mail, HTTP, FTP, remote login, network file systems, finger, Kerberos, and SNMP. Some domains may share trust relationships with other domains (either peer-to-peer or hierarchical). Other domains may operate in complete mistrust of all others, providing outgoing connections only, or perhaps severely restricting incoming connections. Users may be local to a single domain or may possess accounts on multiple domains that allow them to freely establish connections throughout the enterprise."</p> <p>p. 354 [SYM_P_0535486]</p> <p>"Service monitors are dynamically deployed within a domain to provide localized real-time analysis of infrastructure (e.g., routers or gateways) and services (privileged subsystems with network interfaces). Service monitors may interact with their environment passively (reading activity logs) or actively via probing to supplement normal event gathering." p. 355 [SYM_P_0535487]</p> <p>"Event Generation and Storage: Audit generation and storage has tended to be a centralized activity, and often gathers excessive amounts of information at inappropriate layers of abstraction. Centralized audit mechanisms place a heavy burden on the CPU and I/O throughput, and simply do not scale well with large user populations. In addition, it is difficult to extend centralized audit mechanisms to cover spatially distributed components such as network infrastructure (e.g., routers, filters, DNS, firewalls) or various common network services."</p>
6	<p>The method of claim 1, wherein the network monitors are deployed at one or more of the following facilities of the enterprise network: {gateways, routers, proxy servers}.</p>	

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
7	The method of claim 1, wherein deploying the network monitors includes placing a plurality of service monitors among multiple domains of the enterprise network.	<p>p. 354 [SYM_P_0535486]</p> <p>"The EMERALD reusable-monitor architecture provides a framework for the organization and coordination of distributed event analysis across multiple administrative domains. EMERALD introduces a service-oriented, layered approach to representing, analyzing, and responding to network misuse. EMERALD's profiling and signature analyses are not performed as monolithic analyses over an entire domain, but rather are deployed sparingly throughout a large enterprise to provide focused protection of key network assets vulnerable to attack. This model leads to greater flexibility whenever the network configuration changes dynamically, and to improved performance, where computational load is distributed efficiently among network resources."</p> <p>p. 363 [SYM_P_0535495]</p> <p>"Domains under EMERALD surveillance are able to detect malicious activity targeted against their network services and infrastructure, and disseminate this information in a coordinated and secure way to other EMERALD monitors (as well as third-party analysis tools) distributed throughout the network. Reports of problems found in one domain can propagate to other monitors throughout the network using the subscription process. EMERALD's subscription-based communication strategy provides mutual authentication between participants, as well as confidentiality and integrity for all intermonitor message traffic (see Section III-F)."</p> <p>p. 363 [SYM_P_0535495]</p> <p>"Domain-wide analysis forms the second tier of EMERALD's layered network surveillance scheme. A domain monitor is responsible for surveillance over all or part of the domain. Domain monitors correlate intrusion reports disseminated by individual service monitors, providing a domain-wide perspective of malicious activity (or patterns of activity). In addition to domain surveillance, the domain monitor is responsible for reconfiguring system parameters, interfacing with other monitors beyond the domain, and reporting threats against the domain to administrators. Lastly, EMERALD enables enterprise-wide analysis, providing a global abstraction of the cooperative community of domains. Enterprise-layer monitors correlate activity reports produced across the set of monitored domains. Enterprise-layer monitors focus on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, and coordinated attacks from multiple domains against a single domain. Through this correlation and sharing of analysis results, reports of problems found by one monitor may propagate to other monitors throughout the network."</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**"EMERALD 1997"**

'203 Claim number	Claim Term	<p style="text-align: center;"><b>EMERALD 1997</b> (printed publication)</p>
		<p>p. 356 [SYM_P_0535488]</p> <p>"Domain monitors may also operate within an enterprise hierarchy, where they disseminate intrusion reports to enterprise monitors for global correlation. Where trust exists between domains, peer-to-peer subscription provides a useful technique for keeping domains sensitized to malicious activity occurring outside their view.</p> <p>Enterprise-layer monitors attempt to model and detect coordinated efforts to infiltrate domain perimeters or prevent interconnectivity between domains. Enterprise surveillance may be used where domains are interconnected under the control of a single organization, such as a large privately owned WAN. Enterprise surveillance is very similar to domain surveillance: the <i>enterprise monitor</i> subscribes to various domain monitors, just as the domain monitors subscribed to various local service monitors. The enterprise monitor (or monitors, as it would be important to avoid centralizing any analysis) focuses on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, or coordinated attacks from multiple domains against a single domain."</p> <p>p. 363 [SYM_P_0535495]</p>
8	<p>The method of claim 7, wherein receiving and integrating is performed by a domain monitor with respect to a plurality of service monitors within the domain monitor's associated network domain.</p>	<p>"Domain-wide analysis forms the second tier of EMERALD's layered network surveillance scheme. A <i>domain monitor</i> is responsible for surveillance over all or part of the domain. <i>Domain monitors</i> correlate intrusion reports disseminated by individual service monitors, providing a domain-wide perspective of malicious activity (or patterns of activity). In addition to domain surveillance, the domain monitor is responsible for reconfiguring system parameters, interfacing with other monitors beyond the domain, and reporting threats against the domain to administrators. Lastly, EMERALD enables enterprise-wide analysis, providing a global abstraction of the cooperative community of domains. Enterprise-layer monitors correlate activity reports produced across the set of monitored domains. Enterprise-layer monitors focus on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, and coordinated attacks from multiple domains against a single domain. Through this correlation and sharing of analysis results, reports of problems found by one monitor may propagate to other monitors throughout the network."</p> <p>p. 356 [SYM_P_0535488]</p> <p>"Domain monitors may also operate within an enterprise hierarchy, where they disseminate intrusion reports to enterprise monitors for</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**"EMERALD 1997"**

'203 Claim number	Claim Term	<p style="text-align: center;"><b>EMERALD 1997</b> (printed publication)</p>
		<p>global correlation. Where trust exists between domains, peer-to-peer subscription provides a useful technique for keeping domains sensitized to malicious activity occurring outside their view.</p> <p>Enterprise-layer monitors attempt to model and detect coordinated efforts to infiltrate domain perimeters or prevent interconnectivity between domains. Enterprise surveillance may be used where domains are interconnected under the control of a single organization, such as a large privately owned WAN. Enterprise surveillance is very similar to domain surveillance: the <i>enterprise monitor</i> subscribes to various domain monitors, just as the domain monitors subscribed to various local service monitors. The enterprise monitor (or monitors, as it would be important to avoid centralizing any analysis) focuses on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, or coordinated attacks from multiple domains against a single domain."</p> <p>p. 363 [SYM_P_0535495]</p>
9	<p>The method of claim 1, wherein deploying the network monitors includes deploying a plurality of domain monitors within the enterprise network, each domain monitor being associated with a corresponding domain of the enterprise network.</p>	<p>"Domain-wide analysis forms the second tier of EMERALD's layered network surveillance scheme. A <i>domain monitor</i> is responsible for surveillance over all or part of the domain. <i>Domain monitors</i> correlate intrusion reports disseminated by individual service monitors, providing a domain-wide perspective of malicious activity (or patterns of activity). In addition to domain surveillance, the domain monitor is responsible for reconfiguring system parameters, interfacing with other monitors beyond the domain, and reporting threats against the domain to administrators. Lastly, EMERALD enables enterprise-wide analysis, providing a global abstraction of the cooperative community of domains. Enterprise-layer monitors correlate activity reports produced across the set of monitored domains. Enterprise-layer monitors focus on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, and coordinated attacks from multiple domains against a single domain. Through this correlation and sharing of analysis results, reports of problems found by one monitor may propagate to other monitors throughout the network."</p> <p>p. 356 [SYM_P_0535488]</p> <p>"Domain monitors may also operate within an enterprise hierarchy, where they disseminate intrusion reports to enterprise monitors for global correlation. Where trust exists between domains, peer-to-peer subscription provides a useful technique for keeping domains sensitized to malicious activity occurring outside their view.</p> <p>Enterprise-layer monitors attempt to model and detect coordinated efforts to infiltrate domain perimeters or prevent interconnectivity</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**“EMERALD 1997”**

'203 Claim number	Claim Term	<p style="text-align: center;"><b>EMERALD 1997</b> (printed publication)</p>
		<p>between domains. Enterprise surveillance may be used where domains are interconnected under the control of a single organization, such as a large privately owned WAN. Enterprise surveillance is very similar to domain surveillance: the <i>enterprise monitor</i> subscribes to various domain monitors, just as the domain monitors subscribed to various local service monitors. The enterprise monitor (or monitors, as it would be important to avoid centralizing any analysis) focuses on network-wide threats such as Internet worm-like attacks, attacks repeated against common network services across domains, or coordinated attacks from multiple domains against a single domain.”</p> <p>p. 363 [SYM_P_0535495]</p>
11	<p>The method of claim 9, wherein the plurality of domain monitors within the enterprise network establish peer-to-peer relationships with one another.</p>	<p>“Where mutual trust among domains exists, domain monitors may establish peer relationships with one another. Peer-to-peer subscription allows domain monitors to share intrusion summaries from events that have occurred in other domains. Domain monitors may use such reports to dynamically sensitize their local service monitors to malicious activity found to be occurring outside the domain’s visibility. Domain monitors may also operate within an enterprise hierarchy, where they disseminate intrusion reports to enterprise monitors for global correlation. Where trust exists between domains, peer-to-peer subscription provides a useful technique for keeping domains sensitized to malicious activity occurring outside their view.”</p> <p>p. 363 [SYM_P_0535495]</p>
12	<p>An enterprise network monitoring system comprising:  a plurality of network monitors deployed within an enterprise network;  said plurality of network monitors detecting suspicious network activity based on analysis of</p>	<p>See '203 claim 1</p> <p>See '203 claim 1</p> <p>See '203 claim 1</p> <p>See '203 claim 1</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
	network traffic data selected from the following categories: {network packet data transfer commands, network packet data transfer errors, network packet data volume, network connection requests, network connection denials, error codes included in a network packet};	
	said network monitors generating reports of said suspicious activity; and	See '203 claim 1
	one or more hierarchical monitors in the enterprise network, the hierarchical monitors adapted to automatically receive and integrate the reports of suspicious activity.	See '203 claim 1
13	The system of claim 12, wherein the integration comprises correlating intrusion reports reflecting underlying commonalities.	See '203 claim 2

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
14	The system of claim 12, wherein the integration further comprises invoking countermeasures to a suspected attack.	See '203 claim 3
15	The system of claim 12, wherein the plurality of network monitors include an application programming interface (API) for encapsulation of monitor functions and integration of third-party tools.	See '203 claim 4
16	The system of claim 12, wherein the enterprise network is a TCP/IP network.	See '203 claim 5
17	The system of claim 12, wherein the network monitors are deployed at one or more of the following facilities of the enterprise network: {gateways, routers, proxy servers}.	See '203 claim 6
18	The system of claim 12,	See '203 claim 7

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
	wherein the plurality of network monitors includes a plurality of service monitors among multiple domains of the enterprise network.	
19	The system of claim 18, wherein a domain monitor associated with the plurality of service monitors within the domain monitor's associated network domain is adapted to automatically receive and integrate the reports of suspicious activity.	See '203 claim 8
20	The system of claim 12, wherein the plurality of network monitors include a plurality of domain monitors within the enterprise network, each domain monitor being associated with a corresponding domain of the enterprise network.	See '203 claim 9
22	The system of claim 20,	See '203 claim 11

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'203 Claim number	Claim Term	EMERALD 1997 (printed publication)
	wherein the plurality of domain monitors within the enterprise network interface as a plurality of peer-to-peer relationships with one another.	

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
1	A computer-automated method of hierarchical event monitoring and analysis within an enterprise network comprising: deploying a plurality of network monitors in the enterprise network; detecting, by the network monitors, suspicious network activity based on analysis of network traffic data selected from one or more of the following categories: {network packet data transfer commands, network packet data transfer errors, network packet data volume, network connection requests, network connection denials, error codes included in a network packet, network connection acknowledgements, and network packets indicative of well-known network-service protocols}; generating, by the monitors, reports of said suspicious activity; and automatically receiving and integrating the reports of suspicious activity, by one or more hierarchical monitors.	See '203 claim 1  See '203 claim 1  See '203 claim 1  See '203 claim 1
2	The method of claim 1, wherein integrating	See '203 claim 2

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**“EMERALD 1997”**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
	comprises correlating intrusion reports reflecting underlying commonalities.	
3	The method of claim 1, wherein integrating further comprises involving countermeasures to a suspected attack.	See '203 claim 3
4	The method of claim 1, wherein the plurality of network monitors include an API for encapsulation of monitor functions and integration of third-party tools.	See '203 claim 4
5	The method of claim 1, wherein the enterprise network is a TCP/IP network.	See '203 claim 5
6	The method of claim 1, wherein the network monitors are deployed at one or more of the following facilities of the enterprise network: {gateways, routers, proxy servers}.	See '212 claim 8
7	The method of claim 1, wherein at least one of said network monitors utilizes a statistical detection method.	See '212 claim 1
8	The method of claim 1, wherein deploying the network monitors includes placing a plurality of service monitors among multiple domains of the enterprise network.	See '203 claim 7
9	The method of claim 8, wherein receiving and integrating is performed by a domain monitor with respect to a plurality of service monitors within the domain monitor's associated	See '203 claim 8

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
10	network domain. The method of claim 1, wherein deploying the network monitors includes deploying a plurality of domain monitors within the enterprise network, each domain monitor being associated with a corresponding domain of the enterprise network.	See '203 claim 9
12	The method of claim 10, wherein the plurality of domain monitors within the enterprise network establish peer-to-peer relationships with one another.	See '203 claim 11
13	An enterprise network monitoring system comprising: a plurality of network monitors deployed within an enterprise network, said plurality of network monitors detecting suspicious network activity	See '615 claim 1
	based on analysis of network traffic data selected from one or more of the following categories: {network packet data transfer commands, network packet data transfer errors, network packet data volume, network connection requests, network connection denials, error codes included in a network packet, network connection	See '615 claim 1
	acknowledgements, and network packets	

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
	indicative of well-known network-service protocols); said network monitors generating reports of said suspicious activity; and one or more hierarchical monitors in the enterprise network, the hierarchical monitors adapted to automatically receive and integrate the reports of suspicious activity.	See '615 claim 1  See '615 claim 1
14	The system of claim 13, wherein the integration comprises correlating intrusion reports reflecting underlying commonalities.	See '203 claim 2
15	The system of claim 13, wherein the integration further comprises invoking countermeasures to a suspected attack.	See '203 claim 3
16	The system of claim 13, wherein the plurality of network monitors include an application programming interface (API) for encapsulation of monitor functions and integration of third-party tools.	See '203 claim 4
17	The system of claim 13, wherein the enterprise network is a TCP/IP network.	See '203 claim 5
18	The system of claim 13, wherein the network monitors are deployed at one or more of the following facilities of the enterprise network: {gateways, routers, proxy servers}.	See '212 claim 8
19	The system of claim 13, wherein the plurality	See '203 claim 7

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**“EMERALD 1997”**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
	of network monitors includes a plurality of service monitors among multiple domains of the enterprise network.	
20	The system of claim 19, wherein a domain monitor associated with the plurality of service monitors within the domain monitor's associated network domain is adapted to automatically receive and integrate the reports of suspicious activity.	See '203 claim 8
21	The system of claim 13, wherein the plurality of network monitors include a plurality of domain monitors within the enterprise network, each domain monitor being associated with a corresponding domain of the enterprise network.	See '203 claim 9
23	The system of claim 21, wherein the plurality of domain monitors within the enterprise network interface as a plurality of peer-to-peer relationships with one another.	See '203 claim 11
34	A computer-automated method of hierarchical even monitoring and analysis within an enterprise network comprising: deploying a plurality of network monitors in the enterprise network, wherein at least one of the network monitors is deployed at a gateway;	See '615 claim 1  "Service monitors are dynamically deployed within a domain to provide localized real-time analysis of infrastructure (e.g., routers or gateways) and services (privileged subsystems with network interfaces). Service monitors may interact with their environment passively (reading activity logs) or actively via probing to supplement normal event gathering." p. 355 [SYM P_0535487]

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
		<p>"Event Generation and Storage: Audit generation and storage has tended to be a centralized activity, and often gathers excessive amounts of information at inappropriate layers of abstraction. Centralized audit mechanisms place a heavy burden on the CPU and I/O throughput, and simply do not scale well with large user populations. In addition, it is difficult to extend centralized audit mechanisms to cover spatially distributed components such as network infrastructure (e.g., routers, filters, DNS, firewalls) or various common network services."</p> <p>p. 354 [SYM_P_0535486]</p>
	detecting, by the network monitors, suspicious network activity based on analysis of network traffic data;	See '615 claim 1
	generating, by the monitors, reports of said suspicious activity; and	See '615 claim 1
	automatically receiving and integrating the reports of suspicious activity, by one or more hierarchical monitors.	See '615 claim 1
35	The method of claim 34, wherein said integrating comprises correlating intrusion reports reflecting underlying commonalities.	See '203 claim 2
36	The method of claim 34, wherein said integrating further comprises invoking countermeasures to a suspected attack.	See '203 claim 3
37	The method of claim 34, wherein the plurality of network monitors include an API for encapsulation of monitor functions and integration of third-party tools.	See '203 claim 4

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
38	The method of claim 34, wherein said network traffic data is selected from one or more of the following categories: {network packet data transfer commands, network packet data transfer errors, network packet data volume, network connection requests, network connection denials, error codes included in a network packet}.	See '615 claim 1
39	The method of claim 34, wherein said deploying the network monitors includes placing a plurality of service monitors among multiple domains of the enterprise network.	See '203 claim 7
40	The method of claim 39, wherein said receiving and integrating is performed by a domain monitor with respect to a plurality of service monitors within the domain monitor's associated network domain.	See '203 claim 8
41	The method of claim 34, wherein said deploying the network monitors includes deploying a plurality of domain monitors within the enterprise network, each domain monitor being associated with a corresponding domain of the enterprise network.	See '203 claim 9
43	The method of claim 41, wherein the plurality of domain monitors within the enterprise network establish peer-to-peer relationships	See '203 claim 11

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances**  
**"EMERALD 1997"**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
44	<p>with one another.</p> <p>A computer-automated method of hierarchical event monitoring and analysis within an enterprise network comprising:</p> <p>deploying a plurality of network monitors in the enterprise network, wherein at least one of the network monitors is deployed at a router;</p>	<p>See '615 claim 1</p> <p>"Service monitors are dynamically deployed within a domain to provide localized real-time analysis of infrastructure (e.g., routers or gateways) and services (privileged subsystems with network interfaces). Service monitors may interact with their environment passively (reading activity logs) or actively via probing to supplement normal event gathering." p. 355 [SYM_P_0535487]</p> <p>"Event Generation and Storage: Audit generation and storage has tended to be a centralized activity, and often gathers excessive amounts of information at inappropriate layers of abstraction. Centralized audit mechanisms place a heavy burden on the CPU and I/O throughput, and simply do not scale well with large user populations. In addition, it is difficult to extend centralized audit mechanisms to cover spatially distributed components such as network infrastructure (e.g., routers, filters, DNS, firewalls) or various common network services." p. 354 [SYM_P_0535486]</p>
45	<p>detecting, by the network monitors, suspicious network activity based on analysis of the network traffic data;</p> <p>generating, by the monitors, reports of said suspicious activity; and</p> <p>automatically receiving and integrating the reports of suspicious activity, by one or more hierarchical monitors.</p> <p>The method of claim 44, wherein said integrating comprises correlating intrusion</p>	<p>See '615 claim 1</p> <p>See '615 claim 1</p> <p>See '615 claim 1</p> <p>See '203 claim 2</p>

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
46	reports reflecting underlying commonalities. The method of claim 44, wherein said integrating further comprises invoking countermeasures to a suspected attack.	See '203 claim 3
47	The method of claim 44, wherein the plurality of network monitors include an API for encapsulation of monitor functions and integration of third-party tools.	See '203 claim 4
48	The method of claim 44, wherein said network traffic data is selected from one or more of the following categories: {network packet data transfer commands, network packet data transfer errors, network packet data volume, network connection requests, network connection denials, error codes included in a network packet}.	See '615 claim 1
49	The method of claim 44, wherein said deploying the network monitors includes placing a plurality of service monitors among multiple domains of the enterprise network.	See '203 claim 7
50	The method of claim 49, wherein said receiving and integrating is performed by a domain monitor with respect to a plurality of service monitors within the domain monitor's associated network domain.	See '203 claim 8
51	The method of claim 44, wherein said	See '203 claim 9

**EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances  
"EMERALD 1997"**

'615 Claim number	Claim Term	EMERALD 1997 (printed publication)
	deploying the network monitors includes deploying a plurality of domain monitors within the enterprise network, each domain monitor being associated with a corresponding domain of the enterprise network.	
53	The method of claim 51, wherein the plurality of domain monitors within the enterprise network establish peer-to-peer relationships with one another.	See '203 claim 11

**EXHIBIT O**

IN THE UNITED STATES DISTRICT COURT  
FOR THE DISTRICT OF DELAWARE

SRI INTERNATIONAL, INC., a California  
Corporation,

Plaintiff,

v.

INTERNET SECURITY SYSTEMS, INC.,  
a Delaware corporation, INTERNET  
SECURITY SYSTEMS, INC., a Georgia  
corporation, and SYMANTEC  
CORPORATION, a Delaware corporation,

Defendants.

C. A. No. 04-1199-SLR

**SRI INTERNATIONAL'S RESPONSE TO ISS-GA'S FIRST SET OF  
REQUESTS FOR ADMISSIONS [NOS. 1-5]**

Plaintiff SRI International, Inc. ("SRI") responds to Defendant Internet Security  
Systems, Inc.'s, a Georgia corporation, ("ISS-GA") First Set of Requests for Admissions  
as follows:

**GENERAL OBJECTIONS**

1. SRI objects to the Definitions and Instructions to the extent that the  
Definitions and Instructions purport to enlarge the obligations of SRI to respond beyond  
those required by the Federal Rules of Civil Procedure and the Local Rules.
2. SRI incorporates by reference its General Objections to ISS-GA's First Set  
of Interrogatories (Nos. 1-18) as recited in its objections and responses to that set of  
Interrogatories.
3. SRI objects to the definition of "SRI" as overly broad.
4. SRI's investigation and analysis is ongoing. SRI reserves the right to  
modify and/or supplement this response as additional information becomes available.

5. In its responses to specific Requests for Admission, SRI may repeat a general objection for emphasis or some other reason. The failure to repeat any General Objection does not waive such General Objection to that Request.

**RESPONSES**

**REQUEST FOR ADMISSION NO. 1:**

Admit that P. Porras and P. Neumann, "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances" [ISS02892 – ISS02904] was published on October 9, 1997 in the Proceedings of the 20<sup>th</sup> National Information Systems Security Conference (NISSC).

**RESPONSE TO REQUEST FOR ADMISSION NO. 1:**

SRI objects to this request to the extent that it calls for a legal conclusion.

Subject to and without waiving its objections, and based on information presently available to SRI, the request is admitted.

**REDACTED**

**REDACTED**

Dated: March 3, 2006

FISH & RICHARDSON P.C.

By: 

John F. Horvath (#4557)

Horvath@fr.com

FISH & RICHARDSON P.C.

919 N. Market St., Ste. 1100

P.O. Box 1114

Wilmington, DE 19889-1114

Telephone: (302) 652-5070

Facsimile: (302) 652-0607

Howard G. Pollack (CA Bar No. 162897)

Gina M. Steele (CA Bar No. 233379)

Katherine D. Prescott (CA Bar No. 215496)

Michael J. Curley (CA Bar No. 230343)

FISH & RICHARDSON P.C.

500 Arguello St., Ste. 500

Redwood City, CA 94063

Telephone: (650) 839-5070

Facsimile: (650) 839-5071

Attorneys for Plaintiff

SRI INTERNATIONAL, INC.

**CERTIFICATE OF SERVICE**

I hereby certify that on March 3, 2006, I served a copy of the SRI  
INTERNATIONAL'S RESPONSE TO ISS-GA'S FIRST SET OF REQUESTS FOR  
ADMISSIONS [NOS. 1-5] on the following individuals in the manner indicated:

***VIA ELECTRONIC AND U. S. MAIL***

Richard L. Horwitz  
David B. Moore  
Potter Anderson & Corroon LLP  
Hercules Plaza  
1313 North Market Street, 6th Floor  
P.O. Box 951  
Wilmington, DE 19899  
Telephone: 302-984-6000  
Facsimile: 302-658-1192  
Email: [rhorwitz@potteranderson.com](mailto:rhorwitz@potteranderson.com)  
Email: [dmoore@potteranderson.com](mailto:dmoore@potteranderson.com)

Attorneys for  
Defendant/Counterclaim Plaintiffs  
Internet Security Systems, Inc., a  
Delaware corporation, and Internet  
Security Systems, Inc., a Georgia  
corporation

***VIA ELECTRONIC AND U. S. MAIL***

Holmes J. Hawkins, III  
Natasha H. Moffitt  
King & Spalding LLP  
191 Peachtree Street N.E.  
Atlanta, GA 30303-1763  
Telephone: 404-572-4600  
Facsimile: 404-572-5145  
Email: [hhawkins@kslaw.com](mailto:hhawkins@kslaw.com)  
Email: [nmoffitt@kslaw.com](mailto:nmoffitt@kslaw.com)

Attorneys for  
Defendant/Counterclaim Plaintiffs  
Internet Security Systems, Inc., a  
Delaware corporation, and Internet  
Security Systems, Inc., a Georgia  
corporation

***VIA ELECTRONIC AND U. S. MAIL***

Theresa A. Moehlman  
Jeffrey Blake  
Bhavana Joneja  
King & Spalding LLP  
1185 Avenue of the Americas  
New York, NY 10036  
Telephone: 212-556-2100  
Facsimile: 212-556-2222  
Email: [tmoehlman@kslaw.com](mailto:tmoehlman@kslaw.com)  
Email: [jblake@kslaw.com](mailto:jblake@kslaw.com)  
Email: [bjoneja@kslaw.com](mailto:bjoneja@kslaw.com)

Attorneys for  
Defendant/Counterclaim Plaintiffs  
Internet Security Systems, Inc., a  
Delaware Corporation, and Internet  
Security Systems, Inc., a Georgia  
Corporation

***VIA ELECTRONIC AND U.S. MAIL***

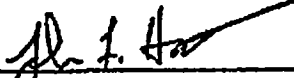
Paul S. Grewal  
Robert M. Galvin, Esq.  
Lloyd R. Day, Jr.  
Day Casebeer Madrid & Batchelder, LLP  
20300 Stevens Creek Boulevard, Suite 400  
Cupertino, California 95014  
Telephone: 408-873-0110  
Facsimile: 408-873-0220  
Email: [pgrewal@daycasebeer.com](mailto:pgrewal@daycasebeer.com)

Attorneys for  
Defendant/Counterclaim Plaintiff  
Symantec Corporation

***VIA ELECTRONIC AND U.S. MAIL***

Richard K. Herrmann  
Morris James Hitchens & Williams LLP  
222 Delaware Avenue, 10th Floor  
P.O. Box 2306  
Wilmington, DE 19899-2306  
Telephone: 302-888-6800  
Facsimile: 302-571-1750  
Email: [rherrmann@morrisjames.com](mailto:rherrmann@morrisjames.com)

Attorneys for  
Defendant/Counterclaim Plaintiff  
Symantec Corporation

  
\_\_\_\_\_  
John E. Horvath

**EXHIBIT P**

IN THE UNITED STATES DISTRICT COURT  
FOR THE DISTRICT OF DELAWARE

SRI INTERNATIONAL, INC., a California  
Corporation,

Plaintiff and  
Counterclaim-Defendant,

v.

INTERNET SECURITY SYSTEMS, INC.,  
a Delaware corporation, INTERNET  
SECURITY SYSTEMS, INC., a Georgia  
corporation, and SYMANTEC  
CORPORATION, a Delaware corporation,

Defendants and  
Counterclaim-Plaintiffs.

C. A. No. 04-1199 (SLR)

**SRI INTERNATIONAL, INC.'S RESPONSE TO SYMANTEC CORPORATION'S  
THIRD SET OF REQUESTS FOR ADMISSION [NOS. 10-89]**

Pursuant to Fed. R. Civ. P. 34, Plaintiff SRI International, Inc. ("SRI"), after reasonable inquiry into the information currently known or readily available, responds to Defendant Symantec Corporation's ("Symantec") Third Set of Requests for Admission as follows:

**GENERAL OBJECTIONS**

1. SRI objects to the Requests for Admission as untimely served. The Requests for Admission were sent by facsimile (after business hours) on March 1, 2006. They were personally served on March 2, 2006. Fact discovery closed on March 31, 2006.
2. SRI objects to the Definitions and Instructions to the extent that the Definitions and Instructions purport to enlarge the obligations of SRI to respond beyond those required by the Federal Rules of Civil Procedure and the Local Rules.
3. SRI objects to the definitions of "SRI" and "Symantec" as overly broad.

4. SRI's investigation and analysis is ongoing. SRI reserves the right to modify and/or supplement this response as additional information becomes available.

5. SRI incorporates by reference the general objections set forth above into the specific objections and response set forth below. SRI may repeat a general objection for emphasis or some other reason. The failure to repeat any general objection does not waive any generation to the request.

**RESPONSES TO REQUESTS FOR ADMISSION**

**REDACTED**

REQUEST FOR ADMISSION NO. 14:

Admit that L.T. Heberlein, B. Mukherjee, K.N. Levitt, "A Method to Detect Intrusive Activity in a Networked Environment," Proc. 14th National Computer Security Conference, pp. 362-371, October 1991 [SYM\_P\_0069355- SYM\_P\_0069365] was published before November 9, 1997.

RESPONSE TO REQUEST FOR ADMISSION NO. 14:

SRI objects to this request to the extent that it calls for a legal conclusion.

Subject to its objections, SRI admits this request.

**REDACTED**

Dated: April 7, 2006

FISH & RICHARDSON P.C.

By: 

Timothy Devlin (#4241)  
John F. Horvath (#4557)  
FISH & RICHARDSON P.C.  
919 N. Market St., Ste. 1100  
P.O. Box 1114  
Wilmington, DE 19889-1114  
Telephone: (302) 652-5070  
Facsimile: (302) 652-0607

Howard G. Pollack (CA Bar No. 162897)  
Gina M. Steele (CA Bar No. 233379)  
Katherine D. Prescott (CA Bar No. 215496)  
Michael J. Curley (CA Bar No. 230343)  
FISH & RICHARDSON P.C.  
500 Arguello St., Ste. 500  
Redwood City, CA 94063  
Telephone: (650) 839-5070  
Facsimile: (650) 839-5071

Attorneys for Plaintiff/Counterclaim Defendant  
SRI INTERNATIONAL, INC.

**CERTIFICATE OF SERVICE**

I hereby certify that on April 7, 2006, I served a copy of SRI  
**INTERNATIONAL, INC.'S RESPONSES TO SYMANTEC'S THIRD SET OF  
REQUESTS FOR ADMISSIONS** to the following in the manner indicated:

**HAND DELIVERY**

Richard L. Horwitz  
Potter Anderson & Corroon LLP  
Hercules Plaza  
1313 North Market Street, 6th Floor  
P.O. Box 951  
Wilmington, DE 19899  
Facsimile: (302) 658-1192

Attorneys for Defendant  
**INTERNET SECURITY SYSTEMS,  
INC.**

**HAND DELIVERY**

Richard K. Herrmann Esq.  
Morris James Hitchens & Williams  
PNC Bank Center  
222 Delaware Avenue, 10th Floor  
P.O. Box 2306  
Wilmington, DE 19899-2306  
Facsimile: (302) 571-1750

Attorneys for Defendant  
**SYMANTEC CORPORATION**

**BY EMAIL & FIRST CLASS MAIL**

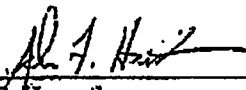
Paul S. Grewal  
Day Casebeer Madrid & Batchelder, LLP  
20300 Stevens Creek Boulevard, Suite 400  
Cupertino, California 95014

Attorney for Defendant  
**SYMANTEC CORPORATION**

**BY EMAIL & FIRST CLASS MAIL**

Holmes Hawkins, III  
King & Spalding  
1180 Peachtree Street, NE  
Atlanta, GA 30309

Attorneys for Defendant  
**INTERNET SECURITY SYSTEMS,  
INC.**

  
\_\_\_\_\_  
John E. Horvath